

Description

System for Loading Executable Code into Volatile Memory in a Downhole Tool

FEDERAL RESEARCH STATEMENT

[0001] This invention was made with government support under Contract No. DE-FC26-01NT41229 awarded by the U.S. Department of Energy. The government has certain rights in the invention.

BACKGROUND OF INVENTION

[0002] This invention relates to a system for loading executable code to volatile memory in a downhole tool, specifically to downhole tools that are part of a downhole tool string. Software for downhole tools on a downhole tool string need to be programmed for numerous conditions. New software may require constant updates to fix errors that result from unforeseen conditions for which the software was not designed. U.S. Pat. No. 6,670,880, which is herein

incorporated by reference, discloses a system for transmitting data through a string of downhole components. In one aspect of the invention in the "880 patent, the system includes first and second magnetically-conductive electrically-insulating elements at both ends of the components. Each element includes a U-shaped trough with a bottom, first and second sides and an opening between the two sides. Electrically conducting coils are located in each trough. An electrical conductor connects the coils in each component. This architecture allows transmission of a signal through the string of downhole components and connects downhole tools to equipment located on the surface. The tools located in the bottom hole assembly and tools located along the string of downhole components require executable code to perform their operations.

[0003] Errors in the executable code of downhole tools complicate the operations of the on-site crew. Errors in the executable code may require portions of the executable code to be rewritten. Often removing the entire downhole tool string is required to access the executable code, which is costly and time consuming. It is not uncommon for a drill string to be 20,000 feet long. Further complicating the

matter is the cost associated with the processors for downhole tools. Due to the extreme conditions downhole, high temperature processors are required for the downhole tools. The transmission system disclosed in the "880 patent is capable of operating many tools at once due to its high speed ability. The combined cost of all of the processors in the downhole tools may add to several million dollars, which increases if errors in the executable code require replacements.

[0004] Attempts at resolving the problems associated with software failures and errors in downhole tools are recognized in the industry. Designing redundant systems with back up processors is one approach. U.S. Pat. No. 4,815,076 discloses a machine implemented process for advisement on several alternatives for recovering from a single or multiple component failures in a distributed process well-site instrumentation logging system. In this process, when a critical part of the downhole system has failed, the role of a failed processor may be shifted to another processor and the software failure may be overcome.

[0005] U.S. Pat. No. 6,061,633 proposes a sonde that includes signal processing means according to a predetermined signal processing program, first memory means for stor-

ing that signal processing program, and second memory means storing a writing program for writing the signal processing program in the first memory means. The sonde's software may be written at the surface, sent over a cable to the sonde, and rebooted from the surface. To avoid removing the sonde every time an error is discovered in the system, the rewritten software is written in electrically erasable programmable read only memory (EEPROM). Due to the limitations of EEPROM, the sonde keeps track of the number of times that the sonde is reprogrammed and after a determined number of reprogrammings, the sonde is retrieved and the EEPROM is replaced. This system reduces the number of times that a downhole tool's software needs to be replaced, but there is still a significant amount of non-volatile memory downhole.

SUMMARY OF INVENTION

[0006] A system for loading executable code into a downhole tool string comprises a surface control unit comprising executable code and a downhole tool string component comprising volatile memory. An integrated downhole network comprises data transmission elements and is in communication with the surface control unit and the

volatile memory. Preferably, the volatile memory is random access memory (RAM). The executable code, stored in the surface control unit, is volatily stored in the downhole tool string component.

[0007] In an embodiment of the present invention, the processor comprises boot memory. The boot memory comprises boot read only memory (ROM), which is the only non-volatile memory used in the processor. The executable code may be requested by the boot memory, and the executable code is written into the volatile memory. Preferably, the executable code is an operating system. Preferably the integrated downhole network comprises network interface modems (NIM) associated with the nodes in the network. In another embodiment of the present invention, there is no boot memory in the processor in the downhole tool. Dual-ported RAM may be housed in the processor. The node and the CPU in the processor may be capable of writing to the dual-ported RAM. A reset mechanism keeps the CPU inactive while the executable code is loading into RAM memory. When the executable code is loaded the reset mechanism activates the CPU and the processor may operate. In other embodiments the RAM has a single port. Preferably, downhole tools are distributed along the

downhole tool string and perform different functions.

Downhole tools string components may comprise one of the following from the group consisting of sensors, motors, jars, seismic sources, seismic receivers, steering elements, hammers, and repeaters.

[0008] Low temperature rated non-volatile memory fails when exposed to high temperature environment found downhole. High temperature non-volatile memory may be expensive. By providing a surface control unit with the executable codes comprising the operating systems for the processors in downhole tools, the processors require less non-volatile memory. However, in the preferred embodiment there is just enough boot memory stored in the non-volatile memory of the downhole processor to execute a program that loads an operating system to the volatile memory of the downhole processor.

[0009] The surface control unit may be a computer. The surface control unit may comprise a connection to a local area network and has access to the internet. On-line access to the surface control unit allows manufacturers to monitor their downhole tools and update the tool's executable code without being on-site.

[0010] The data transmission elements may a magnetically-con-

ductive electrically-insulating material. Preferably, the magnetically-conductive electrically-insulating material is ferrite. Alternatively the data transmission elements may comprise direct electrical contacts. Preferably, the integrated downhole network is capable of transmitting power signals and also capable of transmitting data signals. The integrated downhole network may be capable of transmitting information faster than 20 kilobits per second.

Preferably the integrated downhole network is capable of transmitting data faster than 1 megabit per second. More preferably, the integrated downhole network is capable of transmitting data faster than 10 megabits per second.

[0011] A method for loading executable code to volatile memory in at least one processor in at least one downhole tool located in a downhole tool string comprises the steps of sending the executable code from a surface control unit to a processor over an integrated network and the writing by central processing unit the executable code into volatile memory. The executable code is volatily stored in the downhole tool. The method may further comprise a step of requesting by the processor the executable code from the surface control unit over the integrated downhole network. Another step may include reading by the

processor from boot memory to retrieve an executable code from a surface control unit. In certain embodiments of the present invention, the executable code is temporarily written into random access memory.

BRIEF DESCRIPTION OF DRAWINGS

- [0012] Fig. 1 is a perspective view of an embodiment of a downhole tool string.
- [0013] Fig. 2 is a schematic block diagram of an embodiment of a downhole network.
- [0014] Fig. 3 is a schematic block diagram of an embodiment of a network interface modem.
- [0015] Fig. 4 is a schematic block diagram of an embodiment of a processor in a downhole tool.
- [0016] Fig. 5 is a block diagram of an embodiment of a method for loading an executable code.
- [0017] Fig. 6 is a block diagram of an embodiment of a method for loading an executable code.
- [0018] Fig. 7 is a schematic block diagram of another embodiment of a processor in a downhole tool.
- [0019] Fig. 8 is a block diagram of an embodiment of a method for loading an executable code.

DETAILED DESCRIPTION

[0020] This specification is meant to illustrate the present invention and not limit its scope. Other embodiments of the present invention are possible within the scope and spirit of the claims.

[0021] Fig. 1 shows an embodiment of a downhole tool string 32 in a bore hole. The downhole tool string 32 may be a drill string or it may be part of a production well. A derrick 31 suspends the downhole tool string 32. A data swivel 16 connects the surface control unit 14 to the downhole tool string 32. Components 33 of the downhole tool string 32 may be pipes wherein each pipe comprises a box end and a pin end. The pin end and box end connect together and form a joint 34 in the downhole tool string 32. In the preferred embodiment there are inductive couplers in the pin end and box end of the components 33 of the downhole tool string 32. Alternatively direct electrical couplers may be used.

[0022] Examples of tools on a bottom hole assembly 19 comprise sensors, drill bits, motors, hammers, and steering elements. Examples of tools 18 located along the downhole tool string 32 are nodes, jars 35, seismic sources 36, seismic receivers 37, and sensors, and other tools that aid in the operations of the downhole tool string 32. Different

sensors are useful downhole such as pressure sensors, temperature sensors, inclinometers, thermocouplers, accelerometers, and imaging devices.

[0023] Fig. 2 shows a schematic diagram of an embodiment of integrated downhole network 29. The surface control unit 14 connects to the data swivel 16, which maintains an electric connection to the downhole network 29 integrated into the downhole tool string 32. The nodes 17, in the downhole network 29, may comprise a repeater, at least one NIM 15, a tool port 21, a power source 45, and other electronic components. The integrated downhole network 29 may be capable of transmitting information faster than 20 kilobits per second; preferably, faster than 100 kilobits per second. More preferably, the transmission system may be capable of transmitting data faster than 1 megabit per second. In the preferred embodiment, each downhole component 33 comprises a single coaxial cable connecting transmission elements in the ends of the joints 34 of the downhole tool string 32. Alternatively, the integrated downhole network 29 may comprise additional electrical pathways from the downhole tool 18 to the surface control unit 14. Preferably, the electrical pathway is an electrical conductor capable of transmitting a data signal or a

power signal. The electrical pathway may comprise a coaxial cable, a copper wire, a fiber optics cable, or other type of cable.

[0024] Executable codes may be stored in memory 11 in the surface control unit 14. The surface control 14 unit may be a personal computer. Alternatively, the surface control unit 14 may be connected to a personal computer 12 over a local area network 13. Executable codes are copies of memory stored in non-volatile memory that may be loaded into volatile memory, such as the volatile memory of a node 17, tool 18, or bottom hole assembly 19. Examples of executable codes include software, operating systems, at least part of an operating system, at least one calibration constant, at least one data file, and at least one instruction code. The storage 11 of the surface control unit 14 may comprise non-volatile memory or volatile memory. The non-volatile memory may be associated with a hard drive. The volatile memory may be associated with a removable medium (e.g. floppy disk, zip disk) or in a removable drive (e.g. USB drive, removable hard drive). A removable medium allows the workers to program software away from the on-site location and also provides a back up to any non-volatile memory in the surface control

unit 14. Non-volatile memory in the surface control unit 14 may be a type of ROM, such as EPROM and EEPROM. In the preferred embodiment, the executable codes are stored on a hard drive in the surface control unit 14. Also in the preferred embodiment, the surface control unit 14 comprises a form of ROM and also RAM 25. The ROM may store the executable code for downhole volatile memory in the downhole tools 18.

[0025] The surface control unit 14 also stores data from the downhole tools 18 such as measurements from the sensors. In the preferred embodiment, the surface control unit 14 has a user interface so the on-site crew may access the downhole data.

[0026] A local area network 13 may connect the surface control to computers 12 all around the world, through the internet. Updates may be written by the manufacturer without the manufacturer coming to on-site locations, or without delivering updates to the on-site locations. Updates may fix errors in the operating system or may replace older versions of the tool's software. On-line access allows the manufacturer to monitor their tools 18 while the tools 18 operate downhole. Drill strings are usually rented, so companies want to maximize their drilling time. The op-

erating systems that run the downhole tools 18 may experience situations that were unforeseen during its programming. A quick recovery to errors may increase a company's profit.

[0027] When an error is discovered, the manufacturer may fix the problem by rewriting all or a portion of the program and then sending it through the internet to the surface control unit 14. Other on-sites locations not experiencing a particular error may receive the same updates avoiding the same experience. The manufacturer may have the executable codes stored on their server and the executable codes may be received over the local area network 13. Preferably, the manufacturer backs up the executable codes. Alternatively, the manufacturer's server may be the primary location for the executable codes.

[0028] The downhole tool string 32 may comprises tools 18 made by different manufacturers. The surface control unit 14 may store executable codes for the different tools 18 from different manufactures. Storing the executable codes in the processor 30 of the downhole tools 18 in non-volatile memory is expensive. Loading the executable codes from the surface control unit 14 into downhole volatile memory eliminates the need for most of the non-

volatile memory downhole. Reducing the amount of non-volatile memory downhole greatly reduces costs. One processor in the surface control unit 14 that stores all of the executable codes for the downhole tools 18 on the downhole tool string 32 may be expensive, but the cost of this processor compared to the cost of all of the processors in the downhole tools 18, may be significantly cheaper. Further, the non-volatile memory storage in the surface control unit 14 may not experience the same extreme conditions as the downhole tools 18 and may be stored by a less expensive means.

[0029] The surface control unit 14 comprises a NIM 15. Preferably, the NIM 15 translates the binary form of the executable code to an analogue form to be transmitted downhole via the downhole network 29. Preferably, each downhole tool 18 along the downhole network 29 is also associated with a node 17. Alternatively, the downhole tool 18 may comprise the NIM 15. An embodiment of the node 17 is disclosed in Fig. 3.

[0030] The processing element 46 determines, based off of a packet header, where to send a packet. If the packet header indicates that the packet's address is for an associated tool 18 then the packet is passed to the processor

30 of that downhole tool 18 as shown in Fig. 4. The packet leaves the node 17 through the tool port 21. If the packet header indicates that the packet belongs in another destination, then the processing element 46 sends the packet out through a NIM 15. The network 29 in the preferred embodiment uses QPSK to modulate the packets, but other embodiments include CW, OOK, PCM, FSK, QAM or similar modulation types. Packets in the downhole network 29 may be like any packets known in the art. The network 29 may follow the protocols of collision based networks, token based networks, or other types of networks.

[0031] Fig. 4 shows a schematic block diagram of an embodiment of a processor 30 in a downhole tool 18. A central processing unit (CPU) 23 directs a packet of at least part of an executable code to RAM 25 and the executable code is written into RAM 25. Other volatile memory types may be used in other embodiments of the present invention. The CPU 23 may also interact with an I/O interface 26, which operates actuators 27 and sensors 28. The executable code may be a software executable code for the operations of the tool 18, including the downhole tool's operating system or calibration constants needed to per-

form its operations. The executable code may be a data file. The time of day and other useful types of information may be sent to the processors 30 in downhole tools 18 as well. Other executable codes may include codes for the tool 18 to synchronize its operations with other tools, set parameters, reset registers, read registers, find the status of devices downhole, and perform other requests.

[0032] An embodiment of a method 57 for retrieving calibration constants is illustrated in Fig. 5. The first step 49 comprises the CPU requesting calibration constants from the surface control unit 14. The operating system may indicate to the CPU 23 that calibration constants are needed to set parameters for the downhole tool 18 to operate. During a second step 50 the surface control unit 14 sends the calibration constants to the downhole tool 18 over the downhole network 29. The processing elements 46 direct the packets containing the calibration constants to the correct downhole tool 18. Upon receiving the calibration constants, another step 51 is performed, where the CPU 23 writes the calibration constants into the volatile memory. Preferably, the volatile memory is RAM 25. Once the executable codes are written in RAM 25, the CPU 23 may read from the RAM 25 on how to operate the downhole

tool 18. The CPU 23 sends instructions to the I/O interface 26 that receives data from sensors 28 and sends signals to actuators 27. The data gained by the sensors 28 is sent to the surface control unit 14 where it is stored and easily accessible to the on-site crew.

[0033] In the preferred embodiment, the processors 30 are designed to comprise the least amount of ROM as possible. ROM, EPROM, and EEPROM are heat sensitive, which make using non-volatile memory downhole expensive and difficult. The processors 30 downhole need to be designed to handle high temperatures. High temperature resistant processors may be expensive and fewer high-temperature processors downhole reduce costs for the entire downhole network 29. In the preferred embodiment, the only ROM in the processors 30 of the downhole tool 18 is the boot ROM 24.

[0034] In the preferred embodiment, boot memory 24 is necessary for the processor 30 to operate. When the downhole tool 18 makes electrical contact with a power source, electricity flows into the circuits of the processor 30. The boot ROM 24 is awakened by the electrical current and sends a one-way signal to the CPU 23 requesting the CPU 23 to retrieve an executable code of the processor's oper-

ating system. The CPU 23 requests the executable code from the surface control unit 14. The packets sent by the CPU 23 include an identification of the tool 18 that sent the request. This identification may be a manufacturer number, a serial number, or a model number. The identification tells the surface control unit 14, which operating system's executable code to retrieve. The operating system's executable code is sent back to the CPU 23 and is loaded into the RAM 25.

[0035] An embodiment for a method 58 for correcting errors in downhole tools 18 is shown in Fig. 6. The first step 41 comprises discovering an error in some executable code. An error may be discovered in a downhole tool 18 when the downhole tool 18 doesn't operate properly. The executable code may be analyzed from the surface control unit 14. A next step 42 is rewriting the executable code. The code may be written at or near the drilling site, or it may be written else where and sent over the local area network 13. Step 43 comprises loading the rewritten executable code into the surface control unit 14. This may be done by uploading it over the local area network 13 or from a removable media. The next step 44 starts a re-booting process. The downhole tool 18 is restarted. In the

preferred embodiment, the executable code with the error is erased from RAM 25. When the CPU activates, it reads from boot memory 24 in step 45. The boot memory 24 instructs the CPU 23 to look for the operating system in the surface control unit 14. In step 46, the CPU 23 sends a request to the surface control unit 14 for executable code, which will contain code for at least part of the operating system. In another step 47, the surface control unit 14 sends the rewritten executable code to the CPU 23 over the downhole network 29. The signal sent from the surface control unit 14 is directed to the correct downhole tool 18 by the network 29. In the final step 48, the CPU 23 writes the executable code into RAM 25.

[0036] Updates may be added to the executable code during re-booting. The updates may fix errors in previous executable code. The errors may be syntax errors, semantic errors, or algorithmic errors. When the CPU 23 requests the operating system's executable code during rebooting, the updates are included as part of the executable code and the error should be removed. If the surface control unit 14 did not have the executable codes for the processor 30, gaining access to the downhole tool's software would require removing the downhole tool string 32 from

the borehole. A lot of time is saved by rewriting the software and loading it to the surface control unit 14 and then rebooting the downhole network 29. Preferably, the boot ROM 24 is just large enough to ensure that the executable codes are requested.

[0037] Another advantage to using RAM 25 in downhole tools 18 is that RAM 25 is much faster than forms of non-volatile memory. The CPU 23 operates the downhole tool 18 based off the code written in RAM 25. Tools 18 that respond quicker allow data to be sent to the surface quicker. In some situations, as when a high pressure pocket of gas is hit during drilling, an on-site crew needs a warning as soon as possible. Even a warning that arrives several seconds faster could make a critical difference in emergency situations.

[0038] Fig. 7 shows an embodiment of a processor 30 in a downhole tool 18 that requires no ROM. No ROM may be achieved by using a dual ported RAM 40. However, certain embodiments comprising no non-volatile memory in the processor 30 comprise a single ported RAM 25. In an embodiment where the processor 30 has no non-volatile memory downhole, the executable codes must be sent to the processors 30 from the surface control unit 14. The

node 17 may read off and write to the dual ported RAM 40 in the processor 30 of the downhole tool 18. The CPU 23 of the processor 30 may also read off and write to the dual ported RAM 40. Such a processor may 30 require a design that avoids both the CPU and node 17 from interacting with the dual ported RAM 40 at the same time. A reset mechanism 38 may keep the CPU 23 inactive until the necessary executable codes are loaded into the dual ported RAM 40. When RAM 40 has enough code the reset mechanism 38 activates the CPU 23. In this embodiment the nodes 17 comprise enough intelligence to direct the executable to the correct processors.

[0039] An embodiment of method 59 is shown in Fig. 8. The first step 52 of method 59 describes the booting process once a downhole tool 18 achieves electrical communication with the surface control unit 14. Typically, electrical communication is achieved when the downhole tool 18 is attached to the data swivel 16. A second step 53 comprises the CPU 23 activating and reading from boot memory 24. The boot memory 24 will direct the CPU 23 to look for executable code in the surface control unit 14 and in the next step 54, the CPU 23 requests from the surface control unit 14 executable code over the downhole network

29. During the next step 55, the surface control unit 14 sends the executable code to the CPU 23 over the down-hole network 29. In the final step 56, the CPU 23 writes the executable code into volatile memory. Preferably, the downhole tools 18 are booted when they are electrically connected to the surface control unit 14. During tripping, a top-hole adapter connects to the uppermost component in the downhole tool string 32, which has a connection to the surface control unit 14. The top-hole adapter may be a data swivel 16 or any electrical connection joining the surface control unit 14 and the downhole tool string 32. Preferably, the top-hole adapter is connected to the component first before it is added to the downhole tool string 32. This allows an immediate connection to the tools 18 already downhole and the crew may communicate with the downhole tools 18 immediately. Also as the downhole tool string 32 is lowered further into the established borehole, if there is an emergency, the on-site crew has access to downhole information immediately. In this scenario, any tool 18 that is attached to the component to be added to the downhole tool string 32 begins loading its operating system's executable codes before it is connected to the downhole tool string 32.

[0040] In the preferred embodiment of the present invention, when a tool 18 is added to the downhole tool string 32, the boot ROM 24 may automatically request the executable codes from the surface control unit 14. However, when several sections of pipe are being added or removed, such as during tripping, the electrical connection from the main power source may be lost momentarily. Under these conditions, the downhole tools 18 would be turned on and off every time that connection is lost. Booting and rebooting every tool 18 on the downhole tool string 32 may create packet collisions due to high traffic in the downhole network 29. Preferably every node 17 in the downhole network 29 comprises a power source, which allows the tools 18 to operate in a sleep mode when the connection to the main power source is disrupted. The sleeping mode comprises a low power setting which allows the tools 18 to remain on, but not enough to be operational. The nodes 17 on the downhole tool string 32 may be designed to run in this sleep mode for about 40 days. When a signal from the surface reaches a tool 18 in sleeping mode, the tool 18 wakes up and the tool 18 becomes fully powered for operation. This system saves the tools 18 from rebooting repetitively during activities such

as tripping. There may also be a zero power setting option for when no power is needed in the downhole tool string

32.